AMENDMENTS TO THE CLAIMS

Upon entry of this amendment, the following listing of claims will replace all prior versions and listings of claims in the pending application. Please cancel claims 9 and 10, and add claims 16-18 as follows:

Listing of Claims

1. (Previously Presented) A computer-implemented method for efficiently parsing received data files, comprising:

receiving, by a virtual browser, a data file;

retrieving, by the virtual browser, a stored version of the data file and a syntax tree comprising nodes and tokens representing data within the data file, the tree including at least one static node;

comparing, by a comparison engine in communication with the virtual browser, the stored version of the data file with the received data file to identify non-matching content in the received data file;

parsing, by a parsing engine of the virtual browser, only the non-matching content of the received data file to form at least one subtree comprising nodes and tokens representing the non-matching content of the received data file;

replacing, by the virtual browser, at least one static node of the syntax tree with a token; and

creating, by the virtual browser, a mapping from each token to one of the subtrees.

2. (Canceled)

3. (Canceled)

4. (Previously Presented) The computer-implemented method of claim 1 wherein the data file is a web page.

5. (Previously Presented) The computer-implemented method of claim 1 wherein the data file is an HTML file.

6. (Previously Presented) A method for efficiently parsing web pages, comprising:

receiving, by a virtual browser, a first HTML page;

retrieving, by the virtual browser, a cached version of the HTML page and a syntax tree comprising nodes and tokens representing data within the first HTML page, the tree including at least one static node;

comparing, by a comparison engine in communication with the virtual browser, the cached version of the HTML page with the received HTML page to identify non-matching content in the received HTML page;

parsing, by a parsing engine of the virtual browser, only the non-matching content in the received HTML page to form at least one subtree comprising nodes and tokens representing the non-matching content of the received data file;

replacing, by the virtual browser, at least one static node of the syntax tree with a token; and

creating, by the virtual browser, a mapping from each token to one of the subtrees.

7. (Canceled)

8. (Previously Presented) A method for efficiently parsing HTML pages,

comprising:

receiving, by a virtual browser, a first HTML page;

responsive to a determination that a cached version of the HTML page exists: retrieving, by the virtual browser from a cache, the cached version of the HTML page and a first syntax tree comprising nodes and tokens representing data within the first HTML page, the first tree including at least one static node;

comparing, by a comparison engine in communication with the virtual browser, the cached version of the first HTML page with the received HTML page to identify non-matching content in the received HTML page;

parsing, by a parsing engine of the virtual browser, only the non-matching content to form a subtree;

creating, by the virtual browser, a mapping from a token of the first tree to the subtree;

responsive to a determination that the cached version of the HTML page does not exist:

parsing, by the parsing engine of the virtual browser, the received HTML page to form a second syntax tree comprising nodes and tokens representing the non-matching content of the received data file, the second tree containing at least one static node; and

storing the second tree and the received HTML page in the cache.

9. (Cancelled)

10. (Cancelled)

11. (Previously Presented) A method for efficiently parsing received data files, comprising:

receiving, by a virtual browser, a first data file;

retrieving a stored syntax tree from a cache, the stored syntax tree comprising nodes and tokens, representing data within the first data file and containing at least one static node and at least one token;

retrieving, by the virtual browser, a second data file from the cache, the second data file associated with the first data file;

identifying, by a comparison engine in communication with the virtual browser, non-matching content present only in the first data file;

parsing, by a parsing engine of the virtual browser, only the non-matching content of the first data file to form at least one subtree comprising nodes and tokens representing the non-matching content of the received data file; and

mapping, by the virtual browser, at least one of the tokens to at least one of the subtrees.

12. (Previously Presented) The method of claim 11, further comprising:

responsive to identifying non-matching content present only in the first file:

adding, by the virtual browser, at least one new token to the syntax tree.

13. (Previously Presented) A system for efficiently parsing input data from a plurality of content servers,

comprising:

a virtual browser for retrieving content from content servers;

an identification engine, in communication with the virtual browser for identifying retrieved content;

a cache, in communication with the virtual browser, for storing retrieved content and syntax trees comprising nodes and tokens representing data within the retrieved content;

a comparison engine in communication with the virtual browser, for comparing retrieved content with stored content to identify non-matching content not stored in the cache;

a parsing engine of the virtual browser for parsing only the non-matching content identified by the comparison engine, forming subtrees comprising nodes and tokens representing the non-matching content of the received data file and creating a mapping from new tokens to formed subtrees.

14. (Canceled)

15. (Previously Presented) An intermediary for efficiently parsing received data files transmitted between a client and a server, the intermediary comprising:

a cache storing a version of a data file received from a server and a syntax tree comprising nodes and tokens representing data within the data file, the tree including at least one static node;

a comparison engine comparing the stored version of the data file with the received data file to identify non-matching content in the received data file; and

a virtual browser in communication with the comparison engine, retrieving the stored version of the data file and the syntax tree from the cache, parsing only the non-matching content of the received data file to form at least one subtree comprising nodes and tokens representing the non-matching content of the received data file, replacing at least one static node of the syntax tree with a token, and creating a mapping from each token to one of the subtrees.

16. (New) A method for efficiently parsing received data files, the method comprising:

determining, by a service executing on a computing device, that a received web page comprises an object not stored in a cache;

identifying, by the service responsive to a rule, that the object is to be tracked;

parsing, by the service responsive to identifying that the object is to be tracked, content of the received web page to create an abstract syntax tree;

storing, by the service, the abstract syntax tree and the content to the cache;

determining, by the service, that the object of a second received web page is stored in the cache;

retrieving, by the service responsive to determining that the object is stored in the cache, the abstract syntax tree and the content from the cache;

comparing, by the service, the second received web page to the content to identify non-matching content in the second web page;

parsing, by the service, only the non-matching content to generate a subtree; and

modifying, by the service, the abstract syntax tree to comprise a token mapped to the subtree.

17. (New) The method of claim 16, wherein parsing content of the received web page to create an abstract syntax tree further comprises designating each node in the abstract syntax tree as a static node.

18. (New) The method of claim 16, further comprising identifying by the token dynamic content in the abstract syntax tree.